Amendments to the Specification:

Please replace page 14, line 3-21 with the following amended lines:

The current embodiment of the invention uses a four line communication interface and method of communicating between the FPGA within base station 218 (acting as a "virtual microcontroller" 220 or ICE) and the real microcontroller device under test (microcontroller 232). The four line communication interface is time dependent so that different information can be transferred at different times over a small number of communication lines. Moreover, since the two processors operate in lockstep, there is no need to provide bus arbitration, framing, or other protocol overhead to effect the communication between the microcontroller 232 and the virtual microcontroller 220. This interface is used for, among other things, transferring of I/O data from the microcontroller 232 to the FPGA 220 (since the FPGA emulates only the core processor functions of the microcontroller in the current embodiment). A first interface line ~~(Data1)~~ Data0 is a data line used by the microcontroller 232 to send I/O data to the FPGA based virtual microcontroller 220. This line is also used to notify the FPGA 220 of pending interrupts. This ~~Data1~~ Data0 line is only driven by the real microcontroller 232. A second data line ~~(Data2)~~ (Data1), which is bidirectional, is used by the microcontroller 232 to send I/O data to the FPGA based virtual microcontroller of base station 218. In addition, the FPGA 220 uses the ~~Data2~~ Data1 line to convey

halt requests (i.e., to implement simple or complex breakpoints) to the microcontroller 232.

Please replace page 26, line 3-17 with the following amended lines:

When the above system is operating in lock-step synchronization, the particular state of the microcontroller (settings, register values, memory contents, etc.) cannot be observed while running. Only when the virtual microcontroller 220 and microcontroller 232 are halted can these parameters be readily observed. A halt can be carried out in either of two ways. Either a halt can be implemented as a break within the debug code, or a user can initiate a halt (generally due to belief that there is a problem in operation of the software.) In order to handle the halt functions as well as other functions, base station 218 incorporates a gatekeeper circuit 602 as illustrated in FIGURE 8. Gatekeeper [[618]] 602 can communicate with the host computer 210 via interface 214. Gatekeeper circuit 602 also receives inputs from the bus 226 such as data0 and data1 from data lines 242, as well as clock signals 246 including CCLOCK and HCLOCK. Additionally, gatekeeper 602 receives signals from a breakpoint controller 606 which also forms a part of base station 218 to control programmed breaks in the operational code running on virtual microcontroller 220 and standard microcontroller 232.